Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2 ПО ДИСЦИПЛИНЕ:

ТИПЫ И СТРУКТУРЫ ДАННЫХ

ТЕМА: ЗАПИСИ С ВАРИАНТАМИ. ОБРАБОТКА ТАБЛИЦ

Студент **Поздышев А. В.**

Группа **ИУ7-31Б**

Название предприятия **НУК ИУ МГТУ им. Н. Э. Баумана**

Студент _____**Поздышев А. В.**

Преподаватель _____**Барышникова М. Ю.**

2024

ОГЛАВЛЕНИЕ

ОПИСАНИЕ УСЛОВИЯ ЗАДАЧИ 2
ОПИСАНИЕ ВНУТРЕННИХ СТРУКТУР ДАННЫХ 4
ФУНКЦИИ ПРОГРАММЫ 7
ОПИСАНИЕ АЛГОРИТМА 9
НАБОРЫ ТЕСТОВ 11
МЕТОДИКА ЗАМЕРОВ 12
ЗАМЕРЫ 12
ПАМЯТЬ 13
ОЦЕНКА ЭФФЕКТИВНОСТИ 13
ВЫВОД 14
ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ 14

ОПИСАНИЕ УСЛОВИЯ ЗАДАЧИ

Создать таблицу, содержащую не менее 40 записей (тип – запись с вариантами (объединениями) о машинах. Упорядочить данные в ней по возрастанию ключей, двумя алгоритмами сортировки: пузырек и пузырек с запоминанием индекса последнего обмена. Сортируемое поле – стоимость машины. Применяются два способа сортировки, а) С использованием таблицы, б) с использованием массива ключей. Реализовать возможность добавления и удаления записей в ручном режиме, просмотр таблицы, просмотр таблицы в порядке расположения таблицы ключей. Осуществить вывод списка не новых машин указанной марки с одним предыдущим собственником, отсутствием ремонтов в указанном диапазоне цен. Найти среди них иномарки с поддержкой обслуживания.

Исходные данные

Список машин. Поля каждого элемента расположены в одной строке, оканчивающейся символом переноса строки. Информация о машине: марка, страна, техническая поддержка, стоимость, цвет, состояние.

1. Новое состояние
    a. Гарантия (в годах)
2. Не новое состояние
    a. Год выпуска
    b. Пробег
    c. Кол-во собственников
    d. Кол-во произведенных ремонтов

Способ обращения к программе

Запуск программы осуществляется через консоль: ./app.exe. После чего необходимо ввести путь к файлу, содержащему список машин. В результате пользователю будет представлено меню опций.

Меню программы:

1. Вывести таблицу на экран.
2. Добавить элемент в таблицу.
3. Удалить элемент из таблицы по цене.
4. Вывести таблицу по таблице ключей.
5. Вывести таблицу ключей.
6. Вывести отсортированную таблицу ключей.
7. Отсортировать обычной сортировкой таблицу машин по цене.

8. Отсортировать обычной сортировкой таблицу машин по цене по ключам.
9. Отсортировать ускоренной сортировкой таблицу машин по цене.
10. Отсортировать ускоренной сортировкой таблицу машин по цене по ключам.
11. Вывести измерения сортировок.
12. Поиск старых определенных иномарок с тех. поддержкой с 1 предыдущим владельцем в заданном диапазоне цен без ремонта.
13. Открыть новый файл.
14. Записать таблицу в файл.
15. О программе.
0. Выход.

Описание возможных аварийных ситуаций и ошибок пользователя

Аварийные ситуации:

- Некорректный ввод: Пустое поле ввода (ожидание ввода пользователя);
- Некорректный ввод: Переполнения буфера;
- Некорректный ввод: превышение допустимой длины поля машины;
- Некорректный ввод: символьный ввод в числовые поля машины;
- Некорректный ввод: ввод отрицательных чисел в числовые поля машины;
- Некорректный ввод: ввод числовых значений, выходящих за границы типа числового поля машины;
- Переполнение: Добавление элемента, выходящего за границы массива;
- Опустошение: Удаление всех элементов массива.

ОПИСАНИЕ ВНУТРЕННИХ СТРУКТУР ДАННЫХ

Информация о машине храниться в структуре типа car_t

Листинг структуры car_t:

```
#define MAX_LEN_STR 20

typedef struct car // Структура машины.
{
   char mark[MAX_LEN_STR];

   char country[MAX_LEN_STR];

   char color[MAX_LEN_STR];

   bool garanty;

   unsigned price;

   bool state_new;

   state_t state;

} car_t;
```

1. mark – марка машины.
2. country – страна производства машины.
3. color – цвет машины.
4. garanty – буловая величина, отвечающая за наличие тех. поддержки.
5. price – стоимость машины.
6. state_new – буловая величина, отвечающее за состояние машины.
7. state – объединение структур, содержащих вариантные поля машины.

Вариантные поля, содержащиеся для машины с новым состоянием, хранятся в структуре newstate_t.

```
typedef struct  // Структура нового состояния машины.

{
```

```
    unsigned short serlife;

} newstate_t;
```

serlife – гарантия машины в годах.

Размер структуры 2 байт.

Вариантные поля, содержащиеся для машины с новым состоянием, хранятся в структуре oldstate_t.

```
typedef struct // Структура использованног состояния машины.

{

    unsigned short yearof;

    unsigned mileage;

    unsigned short owners;

    unsigned short repairs;

} oldstate_t
```

1. yearof – год производства машины.
2. mileage – пробег машины.
3. owners – кол-во собственников.
4. repairs – кол-во ремонтов.

Размер структуры 12 байт.

Обе структуры с вариантными полями хранятся в объединении state_t

```
typedef union // Объединение  структур состояния машины.

{
```

```
    newstate_t newstate;

    oldstate_t oldstate;

} state_t;
```

1. newstate – структура содержащая вариант. поля новой машины.
2. oldstate – структура содержащая вариант. поля не новой машины.

Размер объединения 12 байт.

Для представления массива машин с индексами используется типа car_el_t.

```
typedef struct car_t_arr_el // Структура (индекс - машина)

{

    unsigned long index;

    car_t car;

} car_el_t;
```

1. index – индекс машины в таблице.
2. car – структура с полями машины.

Для представления массива ключей таблицы машин применяестя структура ckey_t.

```
typedef struct ckey_t // Структура для таблицы ключей индекс - цена.

{

    size_t index;

    unsigned price;
```

```
} ckey_t;
```

1. index – индекс соответствующий цене машины.
2. price – цена машины (ключ поле).

ФУНКЦИИ ПРОГРАММЫ

Qsort_arr

Заголовок:

```
void qsort_arr(void *base_st, void *base_en, size_t const nmemb, size_t const size, int (*cmp)(void const *,void const *));
```

Сортирует массив типа car_el_t по возрастанию цены машины. Функция принимает указатели на начало и конец массива base_st, base_en, кол-во элементов nmemb, размер элемента size, функцию компаратора типа (void const *,void const *). Функция сортирует массив пузырьком с запоминанием последнего обмена по возрастанию.

Sort_arr

Заголовок:

```
void sort_arr(void *base_st, void *base_en, size_t const nmemb, size_t const size, int (*cmp)(void const *,void const *));
```

Сортирует массив элементов различных типов. Функция принимает указатели на начало и конец массива base_st, base_en, кол-во элементов nmemb, размер элемента size, функцию компаратора типа (void const *,void const *). Функция сортирует массив пузырьком по возрастанию.

Fread_arr_car

Заголовок:

```
int fread_arr_car(FILE *file, car_el_t **base_st, car_el_t **base_en);
```

Функция осуществляет запись элементов типа car_el_t из файловой переменной в соответствующий массив. Функция принимает файловую переменную file, двойные указатели на начало base_st и конец base_en массива типа car_el_t. Возвращает код ошибки.

Add_arr_elem

Заголовок:

```
int add_arr_elem(car_el_t **base_st, size_t *nmemb);
```

Функция добавляет элемент типа car_el_t в массив. Принимает указатели типа car_el_t на начало base_st и указатель на размер массива nmemb. Возвращает код ошибки.

Pop_arr_elembyprice

Заголовок:

```
int pop_arr_elembyprice(car_el_t **base_st, size_t *nmemb, size_t const price);
```

Функция удаляет первое вхождение элемента типа car_el_t по цене из массива. Принимает указатели типа car_el_t на начало base_st и указатель на размер массива nmemb, значение цены price. Возвращает код ошибки.

Print_arr_car

Заголовок:

```
void print_arr_car(car_el_t *base_st, size_t const nmemb);
```

Функция осуществляет вывод таблицы машин из массива типа car_el_t. Принимает указатели типа car_el_t на начало base_st и кол-во элементов nmemb массива.

Frint_arr_car

Заголовок:

```
void fprint_arr_car(FILE *file, car_el_t *base_st, size_t const nmemb);
```

Функция осуществляет вывод в файл таблицы машин из массива типа car_el_t. Принимает файловую переменную file, указатели типа car_el_t на начало base_st и кол-во элементов nmemb массива.

Filter_car_param

Заголовок:

```
int filter_car_param(car_el_t *base_st, size_t const nmemb, car_el_t *arr, size_t *len, char
const *mark, unsigned const price_l, unsigned const price_r);
```

Функция находит старые машины определенной марки с 1 владельцем в заданном диапозоне цен. Принимает указатели типа car_el_t на начало фильтруемого массива base_st, размер массива nmemb массив нужных элементов arr, его длину len, марку машины mark, диапозон цен, price_l и price_r. Возвращает код ошибки.

Print_garanted_car

Заголовок:

```
void print_garanted_car(car_el_t *arr, size_t const len);
```

Функция выводит иномарки с тех. поддержкой. Функция принимает массив arr, размер массива len.

Get_keys

Заголовок:

```
int get_keys(ckey_t **key_st, ckey_t **key_en, car_el_t *base_st, size_t const nmemb);
```

Функция получает массив типа ckey_t (массив ключей) из массива типа car_el_t (таблицы). Функция принимает двойные указатели типа ckey_t на начало key_st и конец key_en массива ключей, указатели типа car_el_t на начало base_st и конец base_en массива машин. Возвращает код ошибки.

Print_arr_withkey

Заголовок:

```
void print_arr_withkey(car_el_t *base_st, car_el_t *base_en, ckey_t *key_st, ckey_t
*key_en);
```

Функция осуществляет вывод таблицы машин по таблице ключей. Функция принимает указатели типа car_el_t на начало base_st и конец base_en массива машин, указатели типа ckey_t на начало key_st и конец key_en массива ключей.

Print_keys

Заголовок:

```
void print_keys(ckey_t *key_st, ckey_t *key_en);
```

Функция осуществляет вывод таблицы ключей машин. Функция принимает указатели типа ckey_t на начало key_st и конец key_en массива ключей.

ОПИСАНИЕ АЛГОРИТМА

Алгоритм считывает данные с файла, затем выдает пользователю выбор действий.

1. Действие 1:
    a. Вывод таблицы на экран.
2. Действие 2:
    a. Осуществляет добавление элемента в конец массива.
    b. Массив обновляется.
    c. Массив ключей обновляется.
3. Действие 3:
    a. Осуществляет удаление первого вхождения элемента по полю “цена”.
    b. Массив обновляется.
    c. Массив ключей обновляется.
4. Действие 4:
    a. Вывод таблицы по таблице ключей.
5. Действие 5:
    a. Вывод таблицы ключей.
6. Действие 6:
    a. Сортирует массив ключей по возрастанию.
    b. Вывод таблицы ключей на экран.
7. Действие 7:
    a. Сортирует массив машин по возрастанию пузырьком.
    b. Замеряет время выполнения сортировки.
    c. Выводит отсортированный массив на экран.
8. Действие 8:
    a. Сортирует массив ключей по возрастанию пузырьком.
    b. Замеряет время выполнения сортировки.
    c. Выводит по отсортированному массиву ключей массив машин на экран.
9. Действие 9:
    a. Сортирует массив машин по возрастанию ускоренным пузырьком.
    b. Замеряет время выполнения сортировки.
    c. Выводит отсортированный массив на экран.
10. Действие 10:
    a. Сортирует массив ключей по возрастанию ускоренным пузырьком.
    b. Замеряет время выполнения сортировки.
    c. Выводит по отсортированному массиву ключей массив машин на экран.
11. Действие 11:
    a. Выводит на экран последние проведенные замеры.
12. Действие 12:

a. Фильтрует массив машин по переданным параметрам
b. Выводит из отфильтрованного массива машины, удовлетворяющие условию “имеется тех. поддержка”.

13. Действие 13:
    a. Открывает новый файл в режиме чтения.
    b. Считывает данные в массивы.
    c. Закрывает файл.
14. Действие 14:
    a. Открывает файл в режиме записи.
    b. Записывает таблицу в файл.
15. Действие 15:
    a. Выводит информацию о программе.
16. Действие 0:
    a. Завершает работу.

НАБОРЫ ТЕСТОВ

Тест	Входные данные	Выходные данные
Некорректный ввод: пустая строка.	-	ERR_IO: Ошибка некорректного ввода
Некорректный ввод: поле марки превышает макс. длину.	Bmvx5serious20002202020202	ERR_IO: Ошибка некорректного ввода
Некорректный ввод: поле страны превышает макс. длину.	Thegreatbritanoftheworld	ERR_IO: Ошибка некорректного ввода
Некорректный ввод: в поле тех. поддержки, строка не yes/no	net	ERR_IO: Ошибка некорректного ввода
Некорректный ввод: в поле стоимости машины	dfsf	ERR_IO: Ошибка некорректного ввода
Некорректный ввод: в поле стоимости машины (отрицательное число)	-123123	ERR_IO: Ошибка некорректного ввода
Некорректный ввод: в поле состояния машины, строка не new/old.	newest	ERR_IO: Ошибка некорректного ввода

Некорректный ввод: поле цвета превышает макс. длину.	totalyellowveryyellowvery	ERR_IO: Ошибка некорректного ввода
Некорректный ввод: в одно из вариантных полей: срок гарантии, год пр., пробег, кол-во соб-ков, кол-во рем-тов (отрицательное число)	-4	ERR_IO: Ошибка некорректного ввода
Некорректный ввод: в одно из вариантных полей: срок гарантии, год пр., пробег, кол-во соб-ков, кол-во рем-тов (отрицательное число)	adasfsa	ERR_IO: Ошибка некорректного ввода
Переполнение: добавление нового элемента, в результате которого происходит переполнение массива.	<новая запись машины>	ERR_OVERFLOW: Ошибка переполнения
Опустошение: удаление последнего элемента массива по цене.	<стоимость машины>	ERR_EMPTY: Ошибка пустого массива
Корректный ввод	<корректная запись>	Таблица + новая запись

МЕТОДИКА ЗАМЕРОВ

Вычисление замеров сортировок проводилось следующим образом: для каждого алгоритма и его размера массива находилось среднее время за 10 замеров.

ЗАМЕРЫ

N, кол-во элементов.	Сортировка пузырьком, мс.	Сортировка пузырьком с ключами, мс.	Ускоренная сортировка пузырьком, мс.	Ускоренная сортировка пузырьком с ключами, мс.
50	0,019	0,017	0,015	0,014
100	0,071	0,065	0,045	0,038

200	0,317	0,225	0,148	0,121
500	1,199	1,088	0,888	0,659
800	3,221	2,644	2,321	1,627
1000	4,516	3,485	3,209	2,359

Графики сортировок находятся в папке проекта.

ПАМЯТЬ

N, кол-во элементов.	Размер таблицы машин, байты	Размер таблицы ключей, байты
50	4800	800
100	9600	1600
200	19200	3200
500	48000	8000
800	76800	12800
1000	96000	16000

ОЦЕНКА ЭФФЕКТИВНОСТИ

Размер массива	Отношение времени сортировки исходной таблицы к таблице ключей	Отношение времени ускоренной сортировки исходной таблицы к таблице ключей
50	1,11	1,07
100	1,1	1,28
200	1,41	1,22
500	1,2	1,35
800	1,23	1,43
1000	1.29	1,38

По итогам проведения замерного эксперимента и получения данных эксперимента можно сделать вывод, что скорость работы при сортировке исходной таблицы таблицей ключей возрастает с кол-вом элементов в

массиве. В среднем метод ключей эффективней в 1.22 раза при обычной сортировке и 1.3 раза – при ускоренной.

ВЫВОД

Структурное представление записи позволяет удобно организовывать данные, содержащие различные типы полей. Для вариантной части записи, в нашем случае это поля, содержащие информацию о состоянии машины, эффективней всего использовать объединение, которое способно сократить кол-во выделяемой памяти под элемент. В моем случае память под вариантные поля равна 12 байт, без объединения – 14 байт. Это позволило сэкономить для 1000 элементов 2000 байт.

Применение дополнительного массива ключей позволяет уменьшить время операций над большими исходными таблицами. Так из приведенных замеров видно, что при увеличении элементов в исходной таблице время сортировки по ключам уменьшается. В среднем до 1000 элементов метод ключей эффективней в 1.22 раза при обычной сортировке и 1.3 раза – при ускоренной.

Если операции над большим кол-вом данных нацелены на малое кол-во полей структуры, то эффективнее оперировать таблицей ключей. Если же операции над большим кол-вом данных нацелены на несколько или более кол-во полей структуры, то удобней оперировать самой таблицей.

ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1.Как выделяется память под вариантную часть записи?

Для вариантной часть используется объединение, где каждое его поле разделяет одну и ту же область памяти. Т. е. такой способ дает возможность минимизировать кол-во выделяемой памяти на структуры.

2.Что будет, если в вариантную часть вести данные, несоответствующие описанным?

Если ввод данных не вернет ошибку, то дальнейшее чтение из этих полей непредсказуемо, возможно аварийное завершение программы.

3.Кто должен следить за правильностью выполнения операций с вариантой частью записей?

Ответственность за выполнения операций возлагается на программиста. Он должен реализовать инфраструктуру, которая позволяет правильно выполнять операции над этими записями.

4.Что представляет таблица ключей, зачем она нужна?

Таблица ключей – структура данных, содержащая идентификаторы к исходной таблице. Таблица ключей занимает меньше памяти, что ускоряет выполнения операций для исходной таблицы, например, ее сортировку.

Для представления больших чисел в памяти компьютера можно использовать структуры, в которой будет храниться мантисса, порядок, знак, цифры числа и другое.

5.В каких случаях эффективнее обрабатывать данные в самой таблице, а когда – использовать таблицу ключей?

Таблица ключей хорошо подходит для операций, в которых часто происходит, сравнение, обмен, присваивание или удаление, так как она сокращает время выполнения этих операций. Для выполнения операций одновременно с большим кол-вом полей эффективней будет работать с самой таблицей.

6.Какие способы сортировки предпочтительнее для обработки таблиц и почему?

Лучше всего для сортировок таблиц следует использовать алгоритмы со сложностью O(n log n), например, сортировка слиянием, для больших массивов она требует меньше памяти и более эффективна. Сортировки, которая не ограничена каким-либо типом данных, быстрее O(n log n) не существует.